

# **HOSPITAL QUEUE MANAGEMENT SYSTEM**

*A Report*

*Submitted in partial fulfilment of the  
Requirements for the completion of*

*COURSE BASED PROJECT*

**BACHELOR OF ENGINEERING  
IN  
INFORMATION TECHNOLOGY**

By

**Akash-1602-24-737-066  
Nitiin-1602-24-737-309**



**Department of Information Technology**

**Vasavi College of Engineering (Autonomous)**

**ACCREDITED BY NAAC WITH 'A++' GRADE.**

**(Affiliated to Osmania University and Approved by AICTE)**

**Ibrahim Bagh, Hyderabad-31**

**2025-2026**

## ABSTRACT

**The Hospital Queue Management System** is a hybrid C-based application designed to replicate core operational workflows found in real-world clinical environments. The system automates patient intake, prioritization, and service ordering through a **Priority Queue** mechanism, implemented at the data-structure level to enforce medical urgency logic. Unlike traditional FIFO queue simulations, this project integrates a **multi-factor triage model** where severity, age, and arrival time collectively determine a patient's effective priority score.

To enhance decision-making, the system incorporates a **Machine Learning Wait-Time Predictor**, developed in Python using a Random Forest regression model. This module analyzes historical patient data to forecast expected waiting durations, enabling the application to provide approximate service-time estimates during registration. The ML component operates independently but interfaces seamlessly with the C program, demonstrating cross-language modularity and practical AI integration.

Architecturally, the project adheres to the **MVC (Model–View–Controller)** paradigm. The Model layer encapsulates patient structures, the priority-queue logic, and file-based persistence; the View layer presents a structured console UI; and the Controller mediates user actions, queue operations, and ML predictions. Additional functional modules handle authentication, timestamp management, and CSV-based data storage, ensuring the system remains maintainable and extensible without codebase restructuring.

Beyond simulating queue operations, this application emphasizes **operational realism**, data-driven prediction, and modular software engineering—making it a significantly more advanced implementation than typical DSA console applications.

## TABLE OF CONTENTS

<b>INTRODUCTION .....</b>	<b>4</b>
1.1 Overview	
1.2 Core Architecture	
1.3 Key Features	
1.4 Data Structures Used	
<b>ALGORITHM .....</b>	<b>5</b>
2.1 Admin Flow Algorithm	
2.2 User Flow Algorithm	
<b>FLOWCHART .....</b>	<b>7</b>
3.1 System Flow	
3.2 Queue Operation Flow	
3.3 ML Prediction Flow	
<b>IMPLEMENTATION .....</b>	<b>8</b>
4.1 MVC Architecture	
4.2 Model Layer	
4.3 View Layer	
4.4 Controller Layer	
4.5 Integration With ML Module	
4.6 File Structure and Modules	
<b>OUTPUTS .....</b>	<b>9</b>
5.1 Adding Patient	
5.2 Viewing Queue	
5.3 Serving Patient	
5.4 ML Wait-Time Prediction	
5.5 Saving & Loading Queue Data	
<b>FUTURE SCOPE .....</b>	<b>20</b>
<b>LEARNING NEW THINGS .....</b>	<b>21</b>

## INTRODUCTION:

- OVERVIEW:

This project is a console-based Hospital Queue Management System developed in C. It automates patient registration and service order using a priority-based approach instead of simple arrival time. The system includes a machine learning module that predicts a patient's approximate waiting time based on severity, age, and historical data.

- Core Architecture:

The system follows the MVC (Model–View–Controller) structure.

Model handles patient records, priority queue logic, and file storage.

View manages all display outputs.

Controller coordinates operations, decision-making, and ML prediction calls.

- Key Features:

Priority queue based on severity, age, and arrival time

ML wait-time prediction using a Random Forest model.

Save and load patient data through CSV files.

Real-time operations: add, view, search, and serve patients.

- Data Structures Used:

Priority queue for patient ordering.

Structs for patient information.

File handling for storing and retrieving data.

- Types of Patients (Severity Levels):

Normal, Serious, and Critical.

- Priority Rules:

Critical patients are served first, followed by Serious, then Normal.

If severity is equal, older patients get preference.

If still tied, the earlier arrival is prioritized.

## Algorithm:

The flow of the program is divided into two modes from the main interface: Admin and User.

### 1.Admin Flow:-

Login: System prompts for a password.

Menu Display: Options to Add Patient, View Queue, Search Patient, Serve Patient, or View Statistics.

Add Patient: Collect patient details -> Calculate priority -> Insert into the Priority Queue in the correct position.

View Queue: Traverse the priority queue and display patients in order of priority.

Search Patient: Traverse the queue to find a patient by ID or name.

Serve Patient: Remove the highest priority patient (front of queue) -> Save record to file.

Statistics: Display total patients added, served, and waiting.

Load Data: System loads existing queue records at admin login.

Save Data: Updated queue is written to CSV after every modification.

Exit: Admin returns to main interface.

## 2. User Flow:-

Initialize: Load previous queue data from CSV.

Registration: Take patient details (name, age, problem, severity).

Queue Insert: Compute priority → insert into Priority Queue at correct position.

ML Prediction: Call Python script to estimate waiting time using severity, age, and arrival.

Display Info: Show predicted waiting time and updated queue position.

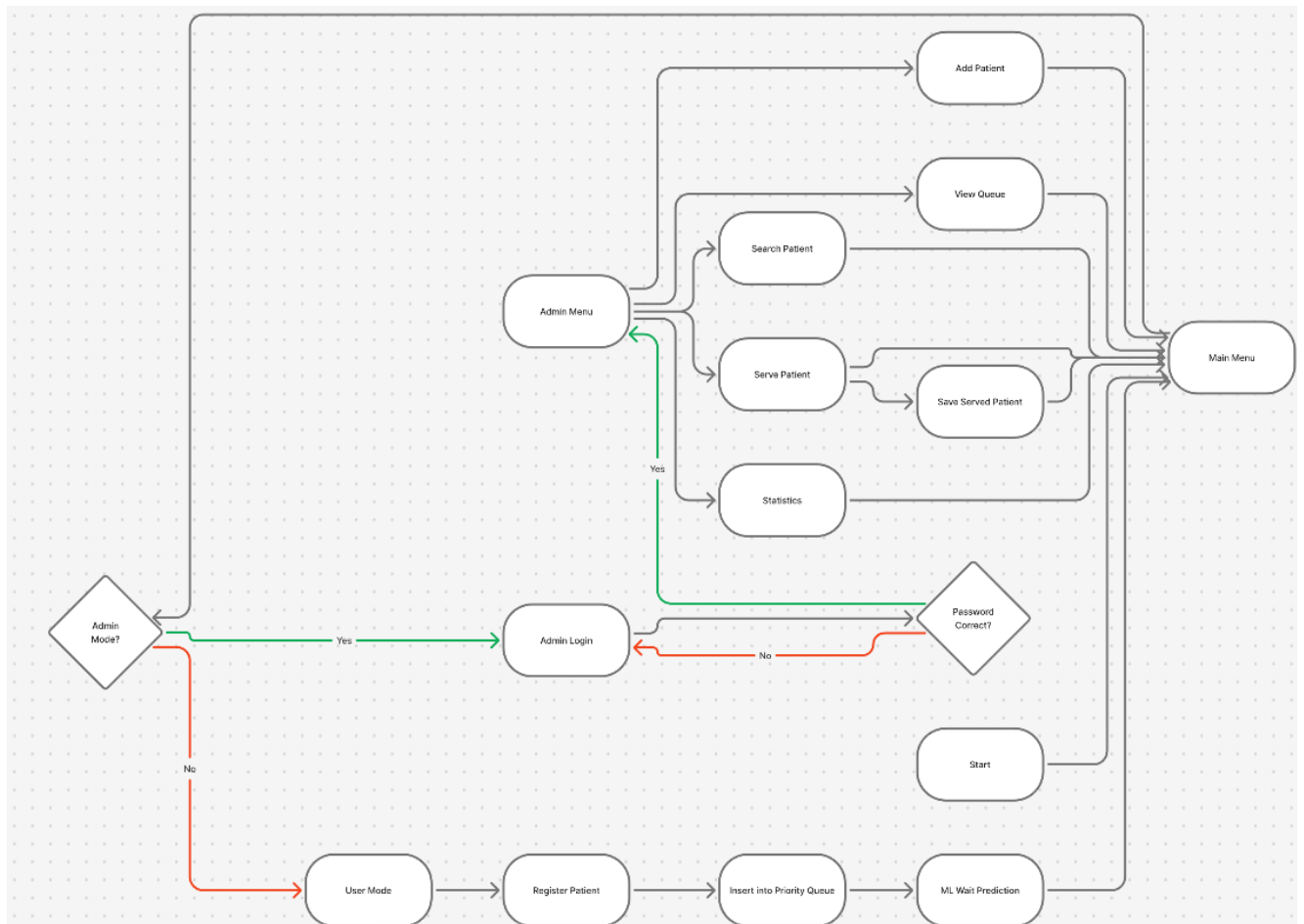
Save Data: Write modified queue to file for persistence.

User Menu: Allow user to view current queue or register another patient.

Error Check: System validates input fields before processing.

Exit: User returns to main interface.

## FLOWCHART:-



## IMPLEMENTATION:-

This project implements a modular, console-based Hospital Queue Management System in C, structured using the Model-View-Controller (MVC) architecture. This design improves clarity, maintainability, and scalability. A priority queue is used to manage patients based on severity, age, and arrival time, while a machine learning module predicts estimated waiting time.

### Architecture Design:-

- **Model:** Handles patient data structures, priority calculation, queue operations, and file persistence functions such as load and save.
- **Controller:** Contains the main program loop, admin/user menus, validates inputs, performs queue operations, and invokes the ML prediction script.
- **View:** Manages all console output including menus, patient details, and messages, keeping display logic separate from core processing.

### Structure Used:-

The application uses C structures to organize data efficiently:

- **Patient Structure:** Stores details like name, age, severity, problem, arrival time, and priority.
- **Queue Node Structure:** Forms the priority queue using linked list nodes.  
Linked list-based priority queue and file handling ensure efficient ordering and data storage.

### File Organization:-

- **model files:** queue.c, patient.c for structures and queue logic.
- **view.c:** Console display functions.
- **controller.c:** Main control flow and operations.
- **main.c:** Entry point of the application.



Outputs:

Execution and Run command:

```
PS C:\Users\AKASH\Downloads\hospital_queue_management_system> python src/tools/wait_predictor.p
train
Trained model saved to C:\Users\AKASH\Downloads\hospital_queue_management_system\src\tools\wait
model.joblib MAE=351.9 sec (5.86 min)
PS C:\Users\AKASH\Downloads\hospital_queue_management_system> gcc -I./src -o hospital_queue_dbg
src/main.c src/controller/controller.c src/auth/auth.c src/model/patient.c src/model/queue.c sr
/view/view.c src/util/time_util.c
PS C:\Users\AKASH\Downloads\hospital_queue_management_system> ./hospital_queue
```

Login interface:

```
=====
HOSPITAL QUEUE MANAGEMENT SYSTEM
=====

Username: █
```

Invalid login:

```
PS C:\Users\AKASH\Downloads\hospital_queue_management_system> ./hospital_queue

=====
HOSPITAL QUEUE MANAGEMENT SYSTEM
=====

Username:
Unknown user
Attempts left: 2
Username: akash
Password:
Invalid credentials
Attempts left: 1
Username: akash
Password:
Invalid credentials
Authentication failed. Exiting.
```

Successful login:

Thereby shows menu

```
=====
HOSPITAL QUEUE MANAGEMENT SYSTEM
=====
```






















Username: akash

Password:

```

MAIN MENU

```

1.  Register Patient
2.  View Waiting List
3.  Call Next Patient
4.  Peek Next Patient
5.  Search Patient
6.  Save Queue
7.  View Statistics
8.  Clear Queue
9.  View Served History
10.  Average Serving Times
11.  Queue Analytics (NEW)
12.  Check Queue Position (NEW)
13.  Predict Wait Time (NEW - ML)
14.  Peak Hours Analysis (NEW)
15.  Staff Performance (NEW)
16.  Emergency Bypass (NEW)
17.  Visual Queue (NEW)
18.  Daily Report (NEW)
19.  Patient Journey (NEW)
20.  System Health Check (NEW)
21.  Exit

Enter choice: █

Register a new patient:

```
Enter choice: 1
Enter name: Rahul sahawariya
Enter age: 23
Enter problem/notes: high fever
Severity (0=Normal,1=Serious,2=Critical): 0
Enter patient's phone number (accepts +91 / 0 / plain): 9899468325
```

✓ Patient registered with ID 72

```
ID: 72
Phone Number: 9899468325
Patient name: Rahul sahawariya
Patient Age: 23
Severity: NORMAL
Arrival: 2025-11-27 23:15:39
Problem: high fever
```

Press Enter to return to menu...█

After successful registration the patient enters into queue.

```
queue.csv X
data > queue.csv > data
1 id,phone,name,age,severity,arrival,problem
2 72,9899468325,Rahul sahawariya,23,0,2025-11-27 23:15:39,high fever
3
```

To view waiting list:


Enter choice: 2

ID	Name	Age	Severity	Arrival	Problem
72	Rahul sahawariya	23	NORMAL	2025-11-27 23:15:39	high fever

Total waiting: 1

To Serve the patient:

Enter choice: 3

 CALLING NEXT PATIENT:

ID: 72

Phone Number: 9899468325


Patient name: Rahul sahawariya

Patient Age: 23

Severity: NORMAL

Arrival: 2025-11-27 23:15:39

Problem: high fever

 Patient serviced successfully

For viewing today's stats

Enter choice: 7

--- Stats ---

Total registered: 1

Total served: 1

Currently waiting: 0

To know average serving time:

Enter choice: 10

Average serving times (min):

NORMAL: 9.3 min (n=20)

SERIOUS: 14.8 min (n=28)

CRITICAL: 17.0 min (n=24)

To search a patient :

Enter choice: 5

Search by (1) ID or (2) name? 2

Enter name or part of name: Rajesh Kumar


[SERVED PATIENTS - MATCHING 'Rajesh Kumar']


ID	Name	Severity	Wait(min)	Phone
1	Rajesh Kumar	CRITICAL	40.00	9876500001
40	Rajesh Kumar	CRITICAL	11.12	9876500001
46	Rajesh Kumar	CRITICAL	7.32	9876500001


To get the peak hour analytics:

Enter choice: 14


 PEAK HOURS ANALYSIS

 Peak Hour: 09:00 (36 patients)

 Staffing Alert: Consider adding staff!

 Hours Summary:

00:00(0) 01:00(0) 02:00(0)  
03:00(0) 04:00(0) 05:00(0)  
06:00(0) 07:00(0) 08:00(1)  
09:00(36) 10:00(15) 11:00(10)  
12:00(5) 13:00(0) 14:00(0)  
15:00(0) 16:00(0) 17:00(0)  
18:00(0) 19:00(0) 20:00(0)  
21:00(0) 22:00(4) 23:00(1)

Press Enter to continue... 

## To get whole serving history:

Enter choice: 9

ID	Phone	Name	Age	Severity	Arrival	Served At	
	Wait(min)	Problem					
1	9876500001	Rajesh Kumar	34	CRITICAL	2025-11-15 09:00:00	2025-11-15 09:40:00	40.00   high fever
2	9123456780	Sneha Sharma	28	SERIOUS	2025-11-15 09:01:10	2025-11-15 09:24:00	23.00   fracture pain
3	9988776655	Mohit Verma	45	NORMAL	2025-11-15 09:02:00	2025-11-15 09:16:00	14.00   cold and cough
4	8765432199	Priya Singh	52	CRITICAL	2025-11-15 09:03:00	2025-11-15 09:35:00	32.00   chest pain
5	7890123456	Anil Patel	60	SERIOUS	2025-11-15 09:04:30	2025-11-15 09:33:00	28.50   shortness of breath
6	9988001122	Aditya Rao	23	NORMAL	2025-11-15 09:05:00	2025-11-15 09:13:00	8.00   headache
7	9770011223	Shruti Kapoor	12	SERIOUS	2025-11-15 09:06:15	2025-11-15 09:23:00	16.75   fever
8	9890012345	Nikhil Das	12	NORMAL	2025-11-15 09:07:00	2025-11-15 09:13:30	6.50   minor cut
9	9001122334	Farah Khan	23	CRITICAL	2025-11-15 09:08:20	2025-11-15 09:40:00	31.50   high fever
10	9011223344	Sameer Ali	45	SERIOUS	2025-11-15 09:09:40	2025-11-15 09:31:30	21.83   allergic reaction
11	9122334455	Divya Roy	38	CRITICAL	2025-11-15 09:10:30	2025-11-15 09:35:00	24.50   urgent care
12	9233445566	Chetan Balan	41	SERIOUS	2025-11-15 09:11:50	2025-11-15 09:30:00	18.17   stomach pain
13	9344556677	Ritu Mehra	29	NORMAL	2025-11-15 09:12:40	2025-11-15 09:20:00	7.33   dizziness
14	9455667788	Pranav Iyer	50	CRITICAL	2025-11-15 09:13:20	2025-11-15 09:44:00	30.67   stroke symptoms
15	9566778899	Kavita Joshi	33	SERIOUS	2025-11-15 09:14:00	2025-11-15 09:36:00	22.00   abdominal pain
16	9677889900	Manish Kulkarni	27	NORMAL	2025-11-15 09:15:30	2025-11-15 09:22:00	6.50   viral fever
17	9788990011	Saira Khan	36	CRITICAL	2025-11-15 09:16:45	2025-11-15 09:40:00	23.33   breathing difficulty
18	9899001122	Vikram Desai	58	SERIOUS	2025-11-15 09:18:00	2025-11-15 09:42:00	24.00   chest discomfort
19	9900112233	Neha Gupta	21	NORMAL	2025-11-15 09:19:10	2025-11-15 09:26:00	6.83   ear pain
20	9012345678	Rahul Sen	44	SERIOUS	2025-11-15 09:20:00	2025-11-15 09:40:00	20.00   checkup
21	9123456701	Deepa Nair	30	CRITICAL	2025-11-15 09:21:25	2025-11-15 09:48:00	26.58   stomach ache
22	9988003344	Arjun Bhat	26	SERIOUS	2025-11-15 09:22:30	2025-11-15 09:43:00	20.50   back pain
23	9770099888	Meera Shah	67	NORMAL	2025-11-15 09:23:40	2025-11-15 09:35:00	11.33   fall injury
24	9891122334	Sameer Khan	55	CRITICAL	2025-11-15 09:24:00	2025-11-15 09:49:00	25.00   high blood pressure
25	9960011223	Alok Mehta	37	NORMAL	2025-11-15 09:25:00	2025-11-15 09:33:30	8.50   ear infection
26	9870012345	Snehal Gupta	29	SERIOUS	2025-11-15 09:25:40	2025-11-15 09:43:00	17.33   migraine
27	9120099888	Varun Shah	50	CRITICAL	2025-11-15 09:26:20	2025-11-15 09:54:00	27.67   acute abdomen
28	9031122334	Leena Bose	44	NORMAL	2025-11-15 09:27:30	2025-11-15 09:38:00	10.50   skin rash
29	9981234567	Aruna Rao	66	CRITICAL	2025-11-15 09:28:10	2025-11-15 09:57:00	28.83   hypertension emergen
30	9892233445	Yusuf Khan	31	SERIOUS	2025-11-15 09:29:00	2025-11-15 09:49:00	20.00   back strain
31	6839581270	Raj Shukla	38	NORMAL	2025-11-15 22:01:13	2025-11-15 22:09:00	7.78   head ache
32	9300664066	Mohit	18	CRITICAL	2025-11-15 22:11:17	2025-11-15 22:23:00	11.50   stress
33	9299408836	Rahul	25	CRITICAL	2025-11-15 22:53:32	2025-11-15 23:05:00	11.50   kidney failure
34	9115458845	Chandrika	26	NORMAL	2025-11-15 22:56:43	2025-11-15 23:05:00	8.28   thyroid
35	9887873428	Raj MK	39	SERIOUS	2025-11-17 11:22:21	2025-11-17 11:36:00	13.63   breathing issue
36	9898766543	Suresh	40	SERIOUS	2025-11-17 11:50:00	2025-11-17 12:00:00	10.00   mental illness

The options on checking on queue:

Enter choice: 2

ID	Name	Age	Severity	Arrival	Problem
74	suresh	12	CRITICAL	2025-11-27 23:39:41	3
73	ramu	34	SERIOUS	2025-11-27 23:38:43	breathing difficulty
75	suresh	12	NORMAL	2025-11-27 23:40:36	skin infection

Total waiting: 3

To get the visualization of queue :

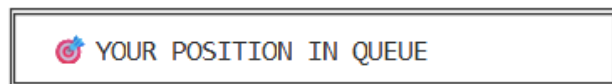
QUEUE ORDER:

- 🔴 [#1] suresh (ID: 74, Age: 12)
- 🟡 [#2] ramu (ID: 73, Age: 34)
- 🟢 [#3] suresh (ID: 75, Age: 12)

To know the position of a patient in the queue:

Enter choice: 12

Enter Patient ID: 73



ID: 73

📍 Position: #1 out of 3

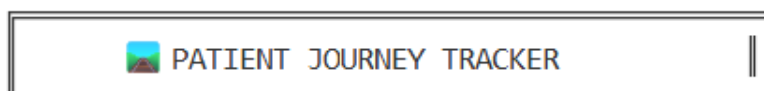
🕒 Estimated Wait: ~5 minutes

🏪 Counter: 2

For Tracking the patients journey , we use

Enter choice: 19

Enter Patient ID: 54



👤 Patient: Aarav Jain (Age: 5)

🟦 Status: SERVED

📍 Arrival Time: 2025-11-18 10:00:00





✅ Service Completed: 2025-11-18 10:10:00





📅 Follow-up: Scheduled for 7 days

To get the daily queue analytics based on severity of patients, we use:

Enter choice: 11

### REAL-TIME QUEUE ANALYTICS

 Total Patients: 3  
 Critical: 1 (33.3%)  
 Serious: 1 (33.3%)  
 Normal: 1 (33.3%)

 Estimated Wait Times:  
 Critical: ~2 min  
 Serious: ~7 min  
 Normal: ~24 min

 System Efficiency: 92.5% (avg throughput)


Press Enter to continue...

For emergency bypass:

Enter choice: 16

### EMERGENCY BYPASS MODE

 CRITICAL Patient ID 74: suresh

 1 critical patient(s) ready for immediate service

Press Enter to continue...



The ML model in the project which is trained on existing data is the unique feature of our project:

```
training.csv X
data > training.csv > data
1 ID,Phone,Name,Age,Severity,Arrival,Served At,Wait(sec),Problem
270 337,9692234793,Rehan Khan,35,0,2025-11-24 12:24:25,2025-11-24 13:18:40,3255,fever
271 338,9675391669,Anil Das,13,1,2025-11-24 13:08:44,2025-11-24 13:25:32,1008,high blood pressure
272 339,8039002960,Chote Khan,27,1,2025-11-24 13:22:22,2025-11-24 13:53:33,1871,body pain
273 340,9183492033,Pooja Kulkarni,9,0,2025-11-24 13:29:04,2025-11-24 13:47:57,1133,muscle pain
274 341,8879715359,Manish Bhat,19,2,2025-11-24 14:02:52,2025-11-24 14:12:56,604,pregnancy checkup
275 342,8625707538,Deepa Bose,23,1,2025-11-24 14:08:38,2025-11-24 14:26:21,1063,fall injury
276 343,9282148741,Arjun Das,42,0,2025-11-24 14:13:25,2025-11-24 15:00:55,2850,shortness of breath
277 344,7514813955,Vikram Balan,38,0,2025-11-24 14:15:20,2025-11-24 14:49:03,2023,allergic reaction
278 345,7916363651,Mohit Mettu,76,1,2025-11-24 14:17:12,2025-11-24 14:28:29,677,shortness of breath
279 346,9071941786,Shruti Kumar,85,0,2025-11-24 14:33:38,2025-11-24 14:59:07,1529,back pain
280 347,8559832672,Sameer Joshi,2,1,2025-11-24 14:48:33,2025-11-24 15:04:52,979,allergic reaction
281 348,7898507463,Sanvi Patel,28,0,2025-11-24 14:57:00,2025-11-24 15:50:57,3237,body pain
282 349,9738092896,Meera Verma,17,0,2025-11-24 15:07:43,2025-11-24 15:47:01,2358,fever
283 350,8472449007,Sneha Nair,50,1,2025-11-24 15:33:20,2025-11-24 15:54:03,1243,high fever
284 351,7512810257,Shravan Nair,50,1,2025-11-24 16:01:07,2025-11-24 16:20:38,1171,chest discomfort
285 352,8144699415,Mohit Sharma,48,0,2025-11-24 16:19:30,2025-11-24 16:54:34,2104,allergic reaction
286 353,8456282067,Ishaan Patel,33,1,2025-11-24 16:24:13,2025-11-24 16:49:59,1546,digestion issue
287 354,9633006316,Priya Sharma,41,0,2025-11-24 16:44:24,2025-11-24 17:39:32,3308,urgent care
288 355,8882874788,Rishi Mehra,27,1,2025-11-24 17:16:30,2025-11-24 17:40:42,1452,headache
289 356,7593866319,Kavya Kulkarni,35,2,2025-11-24 17:18:53,2025-11-24 17:31:31,758,ear infection
290 357,9251839189,Javed Ali,23,0,2025-11-24 17:23:13,2025-11-24 18:17:13,3240,breathing issue
291 358,7803932030,Anil Bhat,9,0,2025-11-24 17:30:37,2025-11-24 17:50:45,1208,stomach ache
292 359,7505350113,Yusuf Joshi,49,1,2025-11-24 18:19:26,2025-11-24 18:46:54,1648,anemia
293 360,8832618262,Nikhil Mehra,24,0,2025-11-24 18:24:52,2025-11-24 19:14:47,2995,chest discomfort
294 361,7516510074,Leena Mehra,44,0,2025-11-24 18:29:33,2025-11-24 19:14:54,2721,pregnancy checkup
295 362,9125587616,Jithesh Sen,47,1,2025-11-24 18:40:31,2025-11-24 19:01:46,1275,fall injury
296 363,8658086061,Nikhil Gupta,50,0,2025-11-24 18:49:46,2025-11-24 19:51:09,3683,checkup
297 364,7670046312,Charan Patel,39,1,2025-11-24 18:54:06,2025-11-24 19:12:23,1097,allergic reaction
298 365,7343516923,Priya Joshi,47,0,2025-11-24 19:35:59,2025-11-24 20:26:02,3003,anemia
299 366,7266336676,Sameer Bhat,69,1,2025-11-24 19:37:02,2025-11-24 19:53:22,980,urgent care
300 367,8449299635,Sahana Singh,16,0,2025-11-24 19:41:16,2025-11-24 20:19:07,2271,high fever
301 368,8254450503,Aarav Khan,47,1,2025-11-24 19:44:59,2025-11-24 20:17:40,1961,fall injury
302
```

- This project includes a Machine Learning module implemented using Python and RandomForestRegressor for predicting patient waiting time.
- The trained model is stored in joblib format and is dynamically called from C at runtime to estimate patient wait time based on severity, age, and arrival patterns derived from historical hospital records


To get the average wait time :

Case 1:

Enter choice: 13


Enter severity (0=Normal, 1=Serious, 2=Critical): 0

Enter age: 15

 AI WAIT TIME PREDICTION (ML)


Severity Level: NORMAL

Age: 15

 Patients Ahead: 3

 ML MODEL PREDICTION:

 PREDICTED WAIT TIME: ~39 minutes


 Model trained on historical records

Case 2:

Enter choice: 13


Enter severity (0=Normal, 1=Serious, 2=Critical): 2

Enter age: 15

 AI WAIT TIME PREDICTION (ML)


Severity Level: CRITICAL

Age: 15

 Patients Ahead: 3

 ML MODEL PREDICTION:

 PREDICTED WAIT TIME: ~11 minutes


 Model trained on historical records

Case 3:

Enter choice: 13


Enter severity (0=Normal, 1=Serious, 2=Critical): 2

Enter age: 15

 AI WAIT TIME PREDICTION (ML)

Severity Level: CRITICAL

Age: 15

 Patients Ahead: 3






















 ML MODEL PREDICTION:

 PREDICTED WAIT TIME: ~11 minutes


 Model trained on historical records

To Exit

MAIN MENU

1.  Register Patient
2.  View Waiting List
3.  Call Next Patient
4.  Peek Next Patient
5.  Search Patient
6.  Save Queue
7.  View Statistics
8.  Clear Queue
9.  View Served History
10.  Average Serving Times
11.  Queue Analytics (NEW)
12.  Check Queue Position (NEW)
13.  Predict Wait Time (NEW - ML)
14.  Peak Hours Analysis (NEW)
15.  Staff Performance (NEW)
16.  Emergency Bypass (NEW)
17.  Visual Queue (NEW)
18.  Daily Report (NEW)
19.  Patient Journey (NEW)
20.  System Health Check (NEW)
21.  Exit

Enter choice: 21

 Exiting... saving queue to data/queue.csv

➤ FUTURE SCOPE:-

- Database Integration :-  
Migrating from CSV file storage to a relational database (MySQL/SQLite) would enable faster lookups, large-scale patient records, multi-department support, and secure authentication for hospital staff.
- Real-Time GUI Dashboard :-  
Porting the system to a graphical interface using GTK, Qt, or a web-based dashboard would allow doctors, nurses, and receptionists to track queue status visually with charts, color codes, and live patient updates.
- Department-Wise Queues :-  
Implementing multiple queues (General, Emergency, Orthopedic, ENT, etc.) would make the system more realistic, allowing separate triage flows while maintaining ML-based predictions for each department.
- Advanced Machine Learning Models :-  
Integrating more features such as doctor availability, current load, hospital timings, and seasonal trends can improve wait-time prediction accuracy. Neural networks or gradient boosting models can be experimented with.
- IoT / RFID Integration :-  
Patient wristbands or token devices could automatically update arrival time, movement, and status without manual entry, enabling smart hospital automation.
- Cloud Synchronization :-  
Storing patient data and queue states on the cloud would allow multi-device access across reception counters, doctor rooms, and mobile applications.

➤ LEARNING NEW THINGS:-

- **Modular Programming :-**  
Understood the advantage of splitting the system into Model, View, and Controller instead of writing everything inside main(). This improved clarity and code organization.
- **Priority Queue Logic :-**  
Learned how to design and implement a priority queue using linked lists, including dynamic insertion based on multiple conditions like severity, age, and arrival time.
- **Machine Learning Integration :-**  
Gained experience in training an ML model using Python (Random Forest), exporting it using joblib, and calling it from C to make real-time predictions.
- **File Handling and Persistence :-**  
Learned how to store and retrieve patient data using CSV files, maintain served history, and ensure data is preserved across program executions.
- **Input Validation and Error Handling :-**  
Practiced validating user entries, handling incorrect inputs, and preventing crashes during patient registration or admin actions.
- **Time and Timestamp Management :-**  
Used time.h and custom utilities to record arrival times, compare timestamps, and prepare proper input features for the ML model.
- **Cross-Language Communication :-**  
Learned how a C program can trigger external Python scripts and read their outputs, enabling hybrid ML-powered systems.
- **Authentication Basics :-**  
Implemented a simple password-protected admin module to restrict sensitive actions like serving patients or viewing statistics.

